LEGIBILITY NOTICE

A major purpose of the Technical Information Center is to provide the broadest dissemination possible of information contained in DOE's Research and Development Reports to business, industry, the academic community, and federal, state and local governments.

Although a small portion of this report is not reproducible, it is being made available to expedite the availability of information on the research discussed herein.

LA-UR -89-1829

LA-UR--89-1829

DE89 012633

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

TITLE: SOFTWARE TOOLS FOR CONTROLLING, DISTRIBUTING, AND INSTALLING SOFTWARE AT LOS ALAMOS NATIONAL LABORATORY

AUTHOR(S): Ron Pfaff, Claudia Sanders, and Marion Cohen

SUBMITTED TO: Supercomputing '89 Conference
Reno, Nevada
November 13-17, 1989

DISCLAIMER

Los Alamos National Laboratory
Los Alamos, New Mexico 87545

Software Tools for Controlling, Distributing, and Installing Software at Los Alamos National Laboratory

by

Ron Pfaff — Computer User Services

Claudia Sanders and Marion Cohen — Computer Documentation

Computing and Communications Division

Los Alamos National Laboratory
Los Alamos, New Mexico 87544

ABSTRACT

This paper focuses on software tools Los Alamos National Laboratory (LANL) uses to control, distribute, and install user level software, such as languages, libraries, and utilities. LANL's computing network is complex enough to represent many, if not all of the problems seen at other sites. The network keeps a system of multi-leveled security partitions separate, while integrating Cray supercomputers with both CTSS and UNICOS operating systems as well as many other types of machines such as VAXes with VMS, SUNs, and Apollos. Change control is the process by which the distribution and installation of new or changed software is controlled. Software tools have been and are being incorporated into the change control procedures to improve efficiency, reliability, and security. Tools for general management are designed to be accessed from any type of computer, while tools for installing files on specific operating systems are designed to be independent. One of the new independent tools is a software package for distributing and installing UNIX™ software. This package, called CCMAKE, could be used as a basis for standardizing a way to assure that all needed files and information are included in a distribution as well as a way to install normal applications, utilities, and libraries. It therefore has the potential to benefit exchanging software between UNIX sites.

UNIX is a Trademark of Bell Laboratories.

Keywords: configuration management, software distribution, software reliability, software installation, change control, documentation.

INTRODUCTION

The Computing and Communications Division (C-Division) at Los Alamos National Laboratory (LANL) supports the Integrated Computing Network (ICN), which is composed of a wide variety of hardware, software, communication facilities, security environments, and computing services offered to users throughout the United States. The ICN integrates large host supercomputers (such as CRAY X-MPs and CRAY Y-MPs), a file server, an output server, a batch server, special-purpose systems for applications (such as CAD/CAM), and general-purpose systems (such as VAXes running VMS or UNIX as well as local area networks of SUNs and Apollos). The common software includes Fortran 77 compilers; four graphics libraries; our Common Los Alamos Mathematical Software (CLAMS); and common utilities to perform such functions as file shipping, graphics file processing, intermachine process-to-process communication, and production job submission. Facilitating and controlling the distribution and installation of changes on systems with a single type of hardware and operating system can be difficult, not to mention the difficulties encountered when performing this function for a system as complex as the ICN.

Change control, a set of procedures based on principles of configuration management and software distribution, is used to control these difficulties. ([1] describes the history and current procedures of change control.) Tools supporting these procedures are needed to maximize security, reliability, and efficiency. The tools and procedures specific to hardware and operating systems can be used independently. For example, tools for distributing and installing UNIX software are independent of the tools used for other operating systems. Generally speaking, tools for general management can be run by any user because the tools are callable from any machine in the ICN by way of centralized server machines. The benefit of this to ICN users is easy addition or deletion of hardware or operating system. The potential benefit to those with similar configuration management and software distribution problems is that the tools deemed useful can easily be separated. In particular, a new tool called CCMAKE may be the basis for a standard way to exchange UNIX (including UNICOS) software between sites.

This paper will discuss the following tools.

- BULLETIN, a tool to generate articles for the *ICN Change Bulletin*. This document announces changes to users.
- ICNNEWS, a tool to announce monthly and interim changes to users online.
- A set of tools to archive C-Division supported software on LANL's massive file storage system, the Common File System (CFS).
- CCGET, a tool acting as a database manager in the respect it minimizes the input necessary to list the full pathnames or get all the desired files from CFS.
- CCMAKE, a tool to distribute and install UNIX software.

This paper concentrates on a subset of tools to give the reader a flavor for the strategies used at LANL. Other change control tools not discussed in this paper include

- a procedure to install public files on CTSS. Among other things, this automatically resizes public file space if needed.
- SAVEPUB/REPUB, a CTSS utility that polls every few minutes for newly installed public files, backs up the newly installed public files on CFS, and is used to reinstall public files on deadstarts.

The tools can be organized into three categories:

- Documenting tools, general tools like BULLETIN and ICNNEWS for documenting and notifying users about software being distributed.
- Archiving tools, general tools to archive files and make files available to users and system managers.
- Packaging and installation tools, tools specific to an operating system. These tools are used by programmers to package software for distribution, by users to pretest software before it is installed on their system, and by system managers to install software on their system. CCMAKE will be described as an example of this type of tool.

The paper continues by discussing each category of tool and how they interface with each other.

DOCUMENTING TOOLS

The role of C-Division's Documentation Group in change control has been to document the software changes accurately and provide that information to users in a timely manner. The primary method of completing that task is through the monthly publication of the *ICN Change Bulletin*. Two tools are used for documenting changes, BULLETIN and ICNHELP.

BULLETIN—Generating Articles

In 1983 the Computer Documentation Group developed and began using a utility called BULLETIN to provide a more efficient way to ensure that all information is included when an article is submitted for publication. What started as a summer project for an undergraduate summer student has become a significant tool for change control. The BULLETIN code is written in the C programming language and is implemented on the UNIX operating system. The utility is menu driven providing the user with options to prepare, edit, preview, and submit articles electronically.

Some problems were encountered with the first version of the utility, but the second version of BULLETIN corrected many of the deficiencies and was more widely accepted by the programmers as a reasonable way to submit articles to the *ICN Change Bulletin*.

Shortly after the second version was installed a set of tools to manage the files change control was responsible for was proposed. The design called for a single point where the information for the current cycle can accumulate. That information is then available to be used by tools as well as by the programmers.

The BULLETIN was chosen to be the single point where information is gathered. The BULLETIN still satisfies the original goal of generating timely,

accurate information for the user, but now it also provides accurate information to be used by other change control tools.

The information is accumulated when a programmer chooses the prepare option in BULLETIN. The programmer is prompted for the name of the software going through change control. With this information the BULLETIN utility accesses database files that contain facts about every piece of software that goes through change control. These files have information about functions, documentation, which operating systems the software is currently on and where files will be placed on CFS.

By using the database files, most of the information requirements for an *ICN Change Bulletin* article is entered automatically. BULLETIN will ask approval before entering some information so that the programmer can provide exception information. For example, if a schedule is different than the standard the programmer can easily change it. The change information, of course, is always entered by the programmer.

The information for CFS is gathered behind the scenes as the programmer prepares his article. This information is appended to the end of the article with comments telling the programmer what it is. The CFS information has two parts, the first is the CFS pathname in which the programmer stores software and the second is the standard CFS pathname into which the change control tools will copy the software as an experimental file(X file). This information has a two fold purpose.

(1) It shows the programmer the correct location to store the software for the beginning of the cycle. That information comes from the database. However, if that information is incorrect the programmer has the opportunity to correct it on the review copy of the article. It is important to note that information will not appear in the final article. Figure two shows a review article.

(2) The second purpose of this information is to provide the information used to automate the placement of the files in the experimental file (called X File Access) location at the proper time.

There are positive impacts for programmers, change control personnel, and users. The programmer knows the storage location for the software. The people responsible for providing that information each month now only need to be sure that database information is current. The user is assured that the CFS pathname that is printed in the *ICN Change Bulletin* is correct because the information in this document is used as input into the tools managing change control CFS files.

The BULLETIN utility plays an important role in the change control cycle by providing the following.

- It ensures accurate, complete information about changes to the user community.
- It helps the programmer provide that information.
- It helps the *ICN Change Bulletin* staff to meet stringent deadlines.
- It provides information for other change control tools in one location.

ICNNEWS—Timely Announcements

Recently the Documentation Group began using the ICNHELP utility to put the *ICN Change Bulletin* online. The information is put into the ICNNEWS topic. To read this online information from any computer in the ICN, a user types

ICNHELP ICNNEWS

This tool, which is based on a VMS HELP library, is organized in a hierarchy of topics and subtopics. The ICNHELP utility plays an important role in the change control cycle by providing timely documentation. Many users of the ICN are not at Los Alamos and receive the *ICN Change Bulletin* through the mail a week after Los Alamos users.

TOOLS TO ARCHIVE SOFTWARE

Four tools used for archiving C-Division supported software on CFS are a parser of *ICN Change Bulletin* articles, GETXFILE, CCMIGRATE, and CCGET.

A Parser of ICN Change Bulletin Articles

As previously discussed, the BULLETIN generates the mapping of the CFS pathname where programmers keep their software and the CFS pathname where the software will be copied as Xfiles (experimental files) as part of the review copy of an *ICN Change Bulletin* article. On the last Wednesday of the month (simultaneously with the final *ICN Change Bulletin* being formatted), a parser creates two files with the following information:

- a mapfile with the mapping of CFS pathnames
- a schedule file with software installation

This eliminates the need for the sometimes late and/or inaccurate hand-carried lists of where the programmers have stored their files.

GETXFILE and CCMIGRATE

On the day before the first Tuesday of the month, GETXFILE will be run in a secure partition. It gets the file of maps from CFS, lets the person running it double check that the maps are correct, then uses the maps to automatically copy the software from the programmer's CFS path to change control's CFS path for Xfiles. If the CFS path for an Xfile does not exist, GETXFILE will create it.

On the day after the third Tuesday of the month, CCMIGRATE will be run in a secure partition. It figures out what Xfiles need to be migrated by using the file of schedules, then migrate the files as follows:

- move the old floor version to a past version.
- move the Xfile version to the new floor version.
- also move the Xfile version to a version with a date associated with it.
- delete the Xfile version.

Both GETXFILE and CCMIGRATE keep a complete audit trail, provide recovery if a run is cut short, and allow for the installation of bug fixes with or without input files. They are written in DEC Command Language (DCL) for the VMS operating system because VMS is what LANL's process server currently runs.

This system results in less work for everyone involved, greater security and reliability, and added functionality in a flexible, modular fashion.

CCGET for Pathname Retrieval

CCGET allows users to retrieve files from CFS as well as list the full CFS pathnames for files of interest. In this way, the utility acts as a database manager. For example, to get all the Xfiles for this month for the partition, hardware, operation system, and software type on which you are logged in, you would type

```
ccget -a
```

CCGET uses a generalized CFS scheme. This scheme provides

- a flexible, uniform way to store software in the partition on which it is to be installed,
- the ability to easily add new hardware types and operating systems as a response to a dynamic environment,
- the means to easily change the type of software that is running on a system, and
- a structure for archived versions of C-Division software.

The CFS scheme is designed to be understandable by using readable directory names. In addition to reliably being able to find a version for any date in the past, this scheme will provide insurance against environmental contamination from such things as viruses, because an entire environment (i.e., all the files installed on a particular machine) can be retrieved. Taking advantage of such things as tarfiles will also provide an opportunity to be complete and archive the source along with what has been installed.

The following example shows a CFS pathname and explains the meaning of each node.

```
/cdiv/PARTITION/HARDWARE/SYSTEM/TYPE
                              /VERSION/FILENAME
```

where

/cdiv
: is a secure root directory.

/PARTITION
: is the partition in which the software is to be installed.

/HARDWARE
: is the type of machine on which the software is to be installed.

/SYSTEM
: is the operating system that is running on that above hardware.

/TYPE
: is the type of software, e.g., X-MP versus Y-MP instructions.

/VERSION
: is an archive version, as opposed to an operating system version.

/FILENAME
: is the name of the file.

An example of a pathname is

/cdiv/r/ymp8/anyctss/xmpinstrcostbl/past/cflib

which is the past version of cflib for generating X-MP instructions with COS internal table format (that is, for compatibility mode) on CTSS on a Y-MP/832 in the secure (r for red) partition.

Our implementation of CCGET is specific to our environment, but the principle can be applied elsewhere. While we have a development version that runs on a UNIX machine, our production version runs on VMS as a FOCUS utility. A FOCUS utility has a front end (installed on each of our supported hardware) that calls a VMS machine (the FOCUS machine) to execute the actual program. A nuance to this design for CCGET is that a procedure with the defaults of the partition,

hardware, operating system, and software type on which CCGET is installed calls this front end. This ensures the defaults match the environment and allows the system manager to easily change the defaults, if so desired, while preserving the advantages of a centrally maintained utility.

CCMAKE FOR UNIX SOFTWARE

CCMAKE is designed to run on any UNIX machine and, therefore, will aid efforts not only on UNICOS but on workstations and distributed processing. There are two sets of tools in the CCMAKE package.

- Tools to help the programmer distribute software.
- Tools to help the system manager install software on a UNIX machine.

The distribution tools help the programmer follow standards, ensure that all needed information and files are packaged together for the system manager, significantly reduce what the programmer would otherwise have to type in, and minimize the knowledge a programmer needs about system management. This set of tools provides standard templates for the following files: a makefile for installation, a copyright file, a readme for installation, and a shell to archive all needed files. The shell and the makefile for installation can be included in the programmer's makefile for compilation. When executed, the shell will call CCMAKE to

(1) update the readme file for installation with
   - a list of all the new files along with their sizes for checksum purposes.
   - a date for a version and a description of the changes.

(2) package all needed files into a tarfile.

(3) stop to let the programmer check to see if all the files were included.

(4) store the tarfile on CFS where the programmer wants.

(5) stop to let the programmer check to see if the file was stored properly.

The system manager's installation tools were designed to be flexible and complete enough to provide all the needed functions for the majority of software and to be reliable with the minimum amount of manual input. As an example, after the defaults are set up, the command

ccmake tar

will get all the Xfiles for the UNIX machine, on which the command is executed, from change control's CFS nodes one at a time. For each file, CCMAKE

(1) checks to see if there is enough space to get the tarfile and untar it.

(2) untars the tarfile in the directory the system manager has designated as the default working directory.

(3) stops for additional instructions. At this point, the system manager can read the readme file and the makefile, skip the installation of this file, or edit files if necessary and continue with the installation. The installation can be continued by manually executing the makefile or by letting CCMAKE execute it.

(4) if the system manager decides to continue the installation and the makefile calls CCMAKE, CCMAKE will
   - install all files with the correct permissions depending on the file type, for example, binary versus library. These files are isolated from the system directories because they are installed in local directories (default specified by the system manager) and linked into the system directories. All the characteristics of the files are shown before and after the installation, so the system manager can make sure that the files were installed correctly.
   - install manual pages if they exist.
   - install extra documentation if it exists.
   - clean up any Xfiles (experimental files).
   - stop for additional instructions. Here the system manager can choose to make more modifications to files and rerun the installation, or to continue.

(5) cleans up all the files that were left behind by the tarfile including the tarfile itself.

(6) sends an electronic mail message to those on the default mailing list that the installation succeeded or failed.

CCMAKE ensures reliability by

- using two test modes (a non-execute mode and a mode that installs the files in a test directory),
- automating back up and back out,
- providing a complete audit trail printed on the screen and accessible for review as the installation progresses,

- providing error control.
- stopping the program in appropriate places to allow the system manager to check how things are going and to allow for tailoring.
- installing Xfiles (experimental files) without disturbing the floor versions.
- sending electronic mail that summarizes successful and unsuccessful installations to those who are interested.

Other features include

- The system manager can override any of the defaults by specifying options on the command line.
- The tarfiles for installation can be gotten from FTP or CFS directories other than the change control nodes.
- The installation of files can be run automatically at any time of the day without the system manager having to monitor the run by installing an entry into the cron file.

We learned some interesting lessons in writing portable programs in shell while developing CCMAKE. For example, CCMAKE was written in Bourne shell to make it easy for system managers tailor the code, but after adding all the features for reliability, the shell was so complicated that this purpose was defeated. Another reason for Bourne shell was our expectation for easy portability (with only possible changes to the pathnames to the UNIX utilities). This was foiled by some of UNIX System V's utilities being a part of the shell and not files in directories. Our Cray analysts were kind enough to correct this situation locally for us on UNICOS by adding files to the appropriate directories. We also found the lack of arrays in the Bourne shell a handicap. On the other hand, performance was not an issue because the normal mode of running CCMAKE was either printing out the audit trail as it occurred (printing to the screen is the performance bottleneck) or running it at night with a cron procedure.

SUMMARY

Change control is the process by which LANL controls and facilitates the introduction of new software into its complex network of computing facilities. The facilities networked together include computers from Cray supercomputers down to PCs, and LANL is commited to providing capabilities for distributed computing. Software tools are necessary to assure that change control is reliable, efucient, and easy to use. This paper has described documenting tools to generate articles and to disseminate articles in a timely fashion, tools to archive and retrieve software from LANL's massive file storage system, and CCMAKE, a package designed to run on any UNIX machine, to distribute and install software in a standard, yet flexible way thus providing the potential to exchange software between different types of UNIX machines as well as the same type of UNIX machines.

REFERENCES

[1] Judy Coleman, Ron Pfaff, Marion Cohen. *Change Control of Software at LANL*, in Proceedings of the Cray User Group Meeting, April 1989.